UNIVERSITÉ PARIS 8 Département d'Informatique

VLISP - AID



Harald WERTZ
Juillet 1977
R.T.–10-77

A I D

«AID» est un ensemble de fonctions LISP destiné à aider le programmeur lors de la conception et la mise au point de ses programmes.

«AID» aide à retrouver l'endroit où a été utilisée telle ou telle variable ou fonction ; montre l'interaction des différentes parties du programme, ... ; et constitue avec DEBUG, EF, ADVISE et BREAK un excellent outil pour la mise au point.

Il existe deux manières de l'utiliser :

- comme programma d'indexation ;
- interactivement, en demandant des renseignements précis.

Exemple : soit une fonction (entre autres)

```
(DE FOO (X Y)
    (IF (NULL Z)
        (CONS (FOO (CAR X))(FOO (CDR Y))) ))
```

à l'exécution la machine donne l'indication suivante

```
Z
ER A8
```

Si vous ne savez plus (ou pas) où vous avez utilisé la variable Z, demandez (DESCR Z), la machine répond

```
xxxxx Z xxxxx
Z IS FREE IN FOO
```

Maintenant il ne reste plus qu'à chercher l'éditeur et debugger le programme .

COMMANDES DE AID

(AID) pour charger le système

(%NOTALL) toutes les fonctions définies à partir de l'instant de l'appel de %NOTALL ne sont pas connues. Pour revenir, il faut taper :

(%ALL) toutes les FEXPR ou EXPR définies à partir de maintenant sont prises en compte par AID (état standard)

(%I) donne l'arbre des appels pour toutes les fonctions que AID connaît et une liste de toutes les variables globales, où elles sont évaluées et où elles sont modifiées. Cette commande fournit la liste de toutes les variables locales, avec indication des fonctions dans lesquelles elles sont déclarées.

(DESCR) pour DESCRibe, livre la description de toute fonction et de toute variable connues par AID

(INDEX) livre les arbres d'appel de toutes les fonctions connues par le système

Si on donne des arguments (en nombre quelconque) à %I, DESCR ou INDEX, la machine ne donne les renseignements que pour ces arguments.

(IF $ftn_1 ... ftn_n$) obligatoire pour que la machine connaisse de ftn_1 à ftn_n, si elles ont été définies soit avant d'utiliser AID, soit en mode %NOTALL

(VARDE) donne la description de toutes les variables connues

(%UNDO) sans argument :

le système perd toute connaissance sur les fonctions et les variables

avec arguments :

le système perd toute connaissance sur les variables et/ou les fonctions données en argument

(%FIN-AID) fait un (%UNDO) et efface toutes les fonctions de AID. Cette fonction aide à quitter DEFINITIVEMENT «AID», et libère tous les doublets utilisés par le système.

ATTENTION

Sous AID, vous pouvez travailler tout comme en LISP (les définitions de fonctions (DE, DF) se révèlent plus longues à l'exécution : le système redéfinit les deux fonctions DE et DF)

AID modifie les P-listes de tous les atomes qu'il connaît (il se produit donc des modifications sur les P-listes).

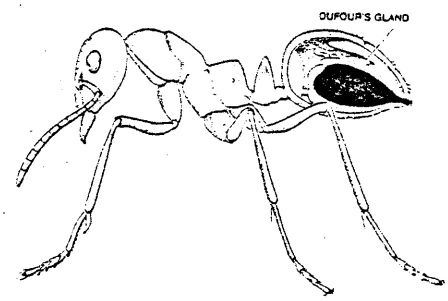
%UNDO efface toute la P-liste des atomes qui sont des variables.

Sauf pour MAP, MAPC, MAPCAR et APPLY, le système ne reconnaît pas les arguments fonctionnels et ne prend pas en compte l'argument d'un QUOTE

Les labels des ESCAPE sont, à la définition, considérés comme des variables et, à l'exit, comme des fonctions.

Les fonctions ou variables aux noms des fonctions standard ne sont pas prises en compte.

Eviter d'utiliser des variables ou fonctions dont le P-NAME commence par le caractère «%». Toutes les variables et fonctions critiques de AID ont un P-NAME commençant par ce caractère.



Slave-making ant

Soit un exemple «bidon»

```
?(DE FØØBAR (X Y)
? (SETQ A (CDR A))
? (CØND
?   ((FØØ X A)(BAR Y (NCØNC B C)))
?   ( T (PRØG (A B)(FØØBAR (CØNS A (LIST B X Y)) B)))))
 FØØBAR
?(DE FØØ (A B)
? (IF (CDR A)(FØØ (FØØBAR (NCØNC1 X A)(FØØ Y (CØPY B))) B)
?     B))
 FØØ
?(DE BAR (A B)
? (IF A B (BAR (FØØBAR A B)(FØØ (SETQ Y B) A))))
 BAR
?(DE CØPY (X)
? (IF X (CØNS (CAR X)(CØPY (CDR X)))))
 CØPY
?(%I)
```

On a défini les 4 fonctions sous AID, regardons d'abord les arbres d'appel

```
 CØPY        CØPY

 BAR         BAR
             FØØ        CØPY
                        FØØ
                        FØØBAR     BAR
                                   FØØ
                                   FØØBAR
             FØØBAR

 FØØ         CØPY
             FØØ
             FØØBAR

 FØØBAR      BAR
             FØØ
             FØØBAR
```

Noter que AID ne reconstruit pas les arbres déjà présentés

puis les variabls

```
 VARIABLE GLØBALES :
 A          IN FØØBAR
 B          IN FØØBAR
 C          IN FØØBAR
 X          IN FØØ
 Y          IN BAR FØØ

 VARIABLE GLØBALES MØDIFIEE :
 A          IN FØØBAR
 B          IN FØØBAR
 C          IN
 X          IN FØØ
 Y          IN BAR
```

C n,est nulle part modifiée

```
 VARIABLE LØCALES :
 A          IN BAR FØØ FØØBAR
 B          IN BAR FØØ FØØBAR
 X          IN CØPY FØØBAR
 Y          IN FØØBAR
 NIL
```

?(DESCR)

****** A ****** D'abord les variables

```
A IS FREE IN FØØBAR
A IS MØDIFIED IN FØØBAR
A IS BØUND IN BAR FØØ FØØBAR
```

****** B ******

```
B IS FREE IN FØØBAR
B IS MØDIFIED IN FØØBAR
B IS BØUND IN BAR FØØ FØØBAR
```

****** C ******

```
C IS FREE IN FØØBAR
```

****** X ******

```
X IS FREE IN FØØ
X IS MØDIFIED IN FØØ
X IS BØUND IN CØPY FØØBAR
```

****** Y ******

```
Y IS FREE IN BAR FØØ
Y IS MØDIFIED IN BAR
Y IS BØUND IN FØØBAR
```

puis les fonctions

****** BAR ******

```
TYPE = EXPR
ARGS = (A B)
VARGLØB = Y
VARGLØB MØDIF = Y
USING = BAR FØØ FØØBAR
USEDBY = BAR FØØBAR
```

****** CØPY ******

```
TYPE = EXPR
ARGS = (X)
USING = CØPY
USEDBY = CØPY FØØ
```

****** FØØ ******

```
TYPE = EXPR
ARGS = (A B)
VARGLØB = X Y
VARGLØB MØDIF = X
USING = CØPY FØØ FØØBAR
USEDBY = BAR FØØ FØØBAR
```

****** FØØBAR ******

```
TYPE = EXPR
ARGS = (X Y)
VARGLØB = A B C
VARGLØB MØDIF = A B
USING = BAR FØØ FØØBAR
```

Demandons l'arbre d'appel de FOO

```
(INDEX FØØ)

 FØØ        FØØBAR     FØØBAR
                       FØØ
                       BAR        FØØBAR
                                  FØØ
                                  BAR
            FØØ
            CØPY       CØPY
 NIL
?(%NØTALL)
 DF
?(DE  PREP (X L)
? (CØND
?   ((NULL L) NIL)
?   ((EQ (CAR L) X)(PREP X (CDR L)))
?   ((CØNS (CAR L)(PREP X (CDR L))))))
 PREP
?(DESCR PREP)
```

Sortons provisoirement de AID et définissons une fonction

```
                     ****** PREP ******

 NIL
?(II PREP)
```

Pas de description, puisque inconnue à AID

```
 PREP       PREP
 NIL
?(DESCR PREP)
```

l'arbre d'appel pour PREP

```
                     ****** PREP ******

 TYPE = EXPR
 ARGS = (X L)
 USING = PREP
 USEDBY = PREP

 NIL
?(DESCR X)

                     ****** X ******

 X IS FREE IN FØØ
 X IS MØDIFIED IN FØØ
 X IS BØUND IN PREP CØPY FØØBAR
 NIL
?(PREP 'X '(X Y X Z))
 (Y Z)
?
```

maintenant il est connu

OK, bonne chance

NOTA : Pour tous renseignements : Harald WERTZ, Département d'Informatique, tél : 374 12 50, poste 299